綴り誤りに対してロバストなローマ字語のマイニング手法

永田　亮†　掛川 淳一†　杉本 洋美††　籔田由己子††
† 兵庫教育大学　†† 教育測定研究所研究開発部
E-mail: †{rnagata,kakegawa}@hyogo-u.ac.jp, ††{sugimoto,yabuta}@jiem.co.jp

1. はじめに

日本人英語学習者の書く英文には，ローマ字で書かれた日本語単語（以後，ローマ字語と表記する）が非常に多い．我々の調査(注1)によると，中学生の書いた英文中では約20%の単語がローマ字語である．ローマ字語が多用される理由として，英語学習者，特に初段階の学習者は語彙が少なく，英単語の代わりにローマ字語を使用することが挙げられる．そのため，名詞だけでなく，形容詞（例：IPPAI(注2)），動詞（例：GANBAROU）など様々な品詞でローマ字語が使用される．また，使用されるローマ字語に綴り誤りが多いことも特徴的である．例えば，前述の英文では，SHSHI，GZYUUNOTOU，MATHYAなどの綴り誤りを含むローマ字語が使用されている（正しい綴りは，4. 評価実験に示す）．

ローマ字語は，様々なタスクでノイズとなり，大きな問題となる．第二言語習得に関する研究では，学習者の特徴を分析する際に，品詞タガーがよく用いられる（例えば，文献[1],[5],[14]）．大部分のローマ字語は未知語となるため品詞タガーの性能に悪影響を及ぼす．その結果，分析にも悪影響を及ぼす．また，同様の理由で，スペルチェカはローマ字語を綴り誤りとして誤検出してしまう．更に，文法の誤り検出[4],[6],[7],[12]では，品詞タガーやチャンカーを利用するため，ローマ字語により検出性能が低下する．

そこで，本論文では，学習者の書いた英文中のローマ字語を自動認識する手法を提案する．ローマ字語と英語の綴りの規則は大きく異なるため，一見，ローマ字語の認識は容易であるように見える．実際には，学習者の英文には綴り誤りが多く，綴りの規則が満たされず，ローマ字語の認識は難しいタスクとなる．綴り誤りに対してロバストな手法とするため，提案手法では少数のルールで補強したクラスタリングアルゴリズムを用いてローマ字語の認識を行う．提案手法には，認識対象の英文と英語の単語リストさえあればローマ字語の認識が行えるという利点がある．

以下，2. で，関連研究について述べる．3. で，提案手法について説明する．4. で，提案手法を評価した実験について述べ，実験結果を考察する．

（注1）：詳細は，4. 評価実験に示す．
（注2）：本論文では，ローマ字語を全て大文字にして表す．

2. 関連研究

著者らが知る限り，日本人英語学習者の書いた英文中のローマ字語を認識する手法は過去に提案されていない．しかしながら，ローマ字語に関連した研究は盛んに行われている．

Transliteration と Back-transliteration [2], [10] は，英語をローマ字表記された日本語へ変換する，またはその逆へ変換する処理を含むことが多い．例えば，Knight と Graehl [10] は，ローマ字表記された日本語を経由して，カタカナ語を英語に変換する手法を提案している．

しかしながら，Transliteration と Back-transliteration は，ローマ字語の認識とは異なるタスクである．Transliteration は，与えられた英語をカタカナ語に変換するタスクである．Back-transliteration は，その逆である．一方，ローマ字語の認識は，日本人英語学習者の書いた英文からローマ字語を発見するタスクである．

ローマ字語の認識に，より関連したタスクとして借用語の認識がある；ローマ字語の認識は，英文中の借用語（ローマ字語）を認識するタスクと捉えることができる．Jeong ら [8] は，韓国語の文書を対象として，借用語と韓国語を区別する手法を提案している．Nwesri ら [13] は，アラビア語の文書から借用語を発見する手法を提案している．Khaltar ら [9] は，日本語の借用語辞書を利用して，モンゴル語中の借用語を発見する手法を提案している．

提案手法は，これらの借用語の認識手法と次の2点について大きく異なる．第一に，提案手法の対象とする文書は，綴り誤りが非常に多い．この綴り誤りのためローマ字語の認識は難いタスクとなる．第二に，上述の手法は，人手で作成した学習データや他の知識源（例えば，文献[9]では，日本語の借用語辞書などを利用する）を必要とし，実装にコストがかかる．一方，提案手法が必要とするものは，認識対象の英文と英語の単語リストのみである．

3. 提案手法

3.1 基本アイデア

ローマ字語と英語とでは綴りの規則が異なるため，単純なパターンマッチでローマ字語と英単語とを区別できる場合が多い．例えば，ローマ字語は，母音または子音 n で終わることから，最後の文字が n 以外の子音である語は，英単語と判断できる．また，英語の単語リストに登録された語は，英語

であることが既知であるので，そもそも認識対象とする必要がない．

しかしながら，パターンマッチと英語の単語リストの組み合わせでは，綴り誤りを含む語を正しく認識できない可能性が高い．例えば，綴り誤りのため子音で終わるローマ字語や子音が連続するローマ字語が学習者の英文には頻出する．また，綴り誤りのある英語は，英語の単語リストに登録されていないので，単語リストでは英語であると認識できない．例えば，4. 評価実験で使用した学習者コーパスでは，英単語 *because* が，綴り誤り *becaus*，*becose*，*becoue*，*becouese*，*becuse*，*becaes*，*becase*，*becaues* として出現する（下線が引かれた語はローマ字語の綴り規則を満たすことを示す）．

綴り誤りに対してロバストな手法とするため，提案手法は英語の単語リストとパターンマッチに加え，クラスタリングアルゴリズムを利用する．まず，英語の単語リストとパターンマッチにより，暫定的なローマ字語リストを作成する．この暫定的なローマ字語リストは，英単語も含むが，ローマ字語を多く含むと期待できる．次に，暫定のローマ字語リストと英語の単語リストから，それぞれ，ローマ字語と英語のセントロイドを求める．最後に，そのセントロイドを初期値とした k-means クラスタリングで，ローマ字語の認識を行う．綴り誤りを含むローマ字語と英語ともに，文字列レベルでは対応する正しい綴りの語に似ていることが多い．したがって，各単語を文字列（具体的には，トライグラム）に基づいたベクトルに変換し，クラスタリングすることで，ローマ字語と英語を正しく認識が行える可能性が高い．

これが，提案手法の基本アイデアである．以下では，提案手法のアルゴリズムを説明する．

3.2 提案手法のアルゴリズム

提案手法のアルゴリズムを以下に示す：

入力： 認識対象文書，英語辞書（英語の単語リスト）

出力： ローマ字語リスト

Step A. 認識対象文書から単語リストを作成

Step B. 作成した単語リストから英語辞書に登録されている語を英語として削除

Step C. パターンマッチにより暫定のローマ字語リストを作成

Step D. 暫定のローマ字語リストと英語辞書から，それぞれ，ローマ字語と英語のセントロイドを計算

Step E. $k = 2$ として k-means クラスタリングで単語リスト中の語をクラスタリング

Step F. ローマ字語のクラスタ中の単語を出力

Step A では，入力として与えられた認識対象文書を単語に分割する．その際に，全ての単語を小文字に変換する．その結果から，単語リストを作成する．

Step B では，得られた単語リストから英語辞書に登録されている語を削除する．この時点で，大部分の正しく綴られた英語が削除される．一方で，綴り誤りを含む英語は単語リストに残る．

Step C では，パターンマッチによりローマ字語の認識を行い，暫定のローマ字語リストを作成する．パターンマッチは次のように行う．まず，各単語を子音母音パターン（CV パターン）に置換する．すなわち，各単語の子音を C に，母音を V に置換する．例えば，ローマ字語：

SAMURAI

は，CV パターン：

CVCVCVV

に，英単語：

fighter

は，CV パターン：

CVCCCVC

に置換できる．置換の例外として，子音 n は，直前に子音がある場合のみに，C に変換する．これは，他の子音と異なる子音 n の振舞（子音 n はローマ字語の語尾の文字として出現可能など）に対応するためである．また，置換の前に，付録に示すルールに用いて，促音と拗音を表す連続した子音の正規化を行う．例えば，連続した二つの子音は一つの子音に置換する（例えば，*tt* を *t* に置換）．この手順によって得られた CV パターンが，正規表現(注 3)：

^[Vn]*(C[Vn]+)*$

にマッチした場合，ローマ字語であると認識する．

Step D では，暫定のローマ字語リストと英語辞書中の各単語をベクトルに変換し，ローマ字語と英語のセントロイドを求める．ベクトルへの変換は，トライグラムに基づいて行う．すなわち，ベクトルの要素はトライグラムに対応し，値は単語中のトライグラムの頻度とする．単語の先頭と末尾を表現するため，先頭と末尾を表す仮想的な文字を付加してトライグラムの抽出を行う（本論文では，先頭と末尾を表す文字として，^と$をそれぞれ利用する）．例えば，ローマ字語：

SAMURAI

からは，トライグラム：

^s^sa sam amu mur ura rai ai$ i$$

が抽出される．したがって，ローマ字語 SAMURAI は，上記のトライグラムに対応する要素の値が 1，それ以外の要素

(注 3)：ここでは，Perl または Java の正規表現に準拠する．

が 0 のベクトルに変換される．ローマ字語のセントロイドは，ローマ字語のリストから得られたベクトルの各要素の平均を取ることで計算できる．英語のセントロイドについても，英語辞書を対象にして同様の手順で計算する．

Step E では，*Step B* の結果得られた単語リスト中の単語をベクトルに変換する．変換手順は，*Step D* と同様である．そのベクトルを，*k*-means クラスタリングでクラスタリングする．ただし，*Step D* のセントロイドを初期値として利用する．

Step F で，ローマ字語のクラスタに属する単語を出力する．また，英語のクラスタに属する単語と *Step B* で，単語リストから削除した単語を出力することで，英単語のリストも出力可能である．

4. 評価実験

4.1 実験条件と実験手順

認識対象として 2 種類のコーパスを利用した．一つめのコーパスとして，中学 2 年生が「冬休み」について書いたエッセイを利用した．二つめのコーパスとして，中学 3 年生が「修学旅行」について書いたエッセイを利用した．表 1 に，各コーパスの概要を示す．表 1 中の “異なり未知語数” とは，次に述べる英語リストに登録されていない語の異なり数を表す．本評価実験では，これらの語のみを評価対象とした．なぜなら，英語リストに登録されている単語は，辞書引きにより容易に英語と認識できるからである．

提案手法で利用する英語リストとして，スペルチェッカ Ispell の英語辞書と British National Corpus [3] で 100 万語あたり 10 回以上出現した単語のリスト [11] を利用した．全体で，19816 語の英語リストが得られた．

比較対象として，3 種類のベースラインを設定した．第一のベースラインでは，上述の英語リストに登録されていない語を全てローマ字語と認識した．第二のベースラインでは，英語リストに登録されていない語を *k*-means クラスタリングで分類した．単語からベクトルへの変換は，提案手法と同じ処理を用いた．初期のセントロイドは，ランダムに選んだ 5 単語から求めた．この手法を，5 回繰り返し，それぞれの結果を平均したものを最終的な評価結果とした．第三のベースラインは，パーターンマッチに基づいた手法とした．このベースラインは，提案手法のアルゴリズムの *Step C* で得られた暫定のローマ字語リストを出力し終了する手法に等しい．

評価尺度として，認識率，認識精度，*F*-measure を用いた．認識率と認識精度は，それぞれ，

$$R = \frac{\text{正しくローマ字語と認識した数}}{\text{認識対象中の異なりローマ字語数}} \quad (1)$$

と，

$$P = \frac{\text{正しくローマ字語と認識した数}}{\text{ローマ字語と認識した数}} \quad (2)$$

で定義した．*F*-measure は，認識率と認識精度を用いて，

$$F = \frac{2RP}{R+P} \quad (3)$$

で定義した．

4.2 評価結果と考察

表 2 と表 3 に，評価結果を示す．表中の，辞書ベース，*k*-means，パターンマッチは，それぞれ，第一，第二，第三のベースラインを表す．

表 2 と表 3 から，英語辞書に登録されていない単語をローマ字語と認識する手法は，認識性能が非常に悪いことがわかる．これは綴り誤りを含む英単語をローマ字語と誤認識するためである．

k-means クラスタリングに基づいたベースラインも，同様に認識性能が低い．*k*-means クラスタリングの結果得られたクラスタに属する単語を分析したところ，ローマ字語の認識という観点からは，意味のないクラスタが得られていることが明らかとなった．得られたクラスタの一例として，動名詞／現在分詞のクラスタ（すなわち *ing* で終わる語のクラスタ）とそれ以外の語から成るクラスタが挙げられる．この結果は，初期のセントロイドをランダムに設定した場合，一つのクラスタで全ての英単語をカバーすることは困難であることを示唆している．

他の二つのベースラインとことなり，パターンマッチに基づいた手法は高い認識性能を示している．これは，ローマ字語と英語では，綴りの規則が大きく異なることに基因している．しかしながら，この手法の性能限界として，ローマ字語，英語ともに，綴り誤りを含む語を正しく認識できない(注 4)．したがって，綴り誤りを含む語において，改善の余地がある．

提案手法は，パターンマッチに基づいた手法を更に改善している．単純な *k*-means クラスタリングに基づいた手法と異なり，初期のセントロイドの計算に，パターンマッチに基づいた手法で得られた認識結果を利用している．そのため，提案手法で最終的に得られたセントロイドは，ローマ字語と英語の特徴を良く表していた．例えば，ローマ字語のセントロイドに特徴的なトライグラムとして母音で終わるもの（例：*i$$*）が確認できた．また，英語に特徴的なトライグラムとして，接尾辞（例：*ed$*）やシラブル（例：*ble*）が確認できた．その結果，他の手法に比べ良い認識性能が得られた．興味深いことに，人でも判断に困るローマ字語を正しく認識できた．例えば，1. で例として示した SHSHI，GZYUUNOTOU，MATHYA を正しくローマ字語と認識できた（正しい綴りは，それぞれ，SUSHI，GOZYUUNOTOU，MATTYA）．これは，認識対象とした英文中に，対応する正しい綴りの語が存

(注 4)：正確には，綴り誤りによってローマ字語の綴りの規則を満たさないローマ字語およびローマ字語の綴りの規則を満たす英語を誤認識する．例えば，SAMURAI を SOMURAI とした場合いは，正しくローマ字語と認識できる．

表 1： 評価実験に用いたコーパスの概要

コーパス	総文数	総単語数	異なり語数	異なり未知語数	異なりローマ字語数
中 2	9928	56724	1675	1040	275
中 3	10441	60546	2163	1334	500

在し，その語の特徴が初期のセントロイドによって捉えられたためである．

表 2： 中学 2 年生の英文に対する認識結果

Method	R	P	F
辞書ベース	**1.00**	0.268	0.423
k-means	0.737	0.298	0.419
パターンマッチ	0.898	0.737	0.810
提案手法	0.855	**0.799**	**0.826**

表 3： 中学 3 年生の英文に対する認識結果

Method	R	P	F
辞書ベース	**1.00**	0.382	0.553
k-means	0.736	0.368	0.490
パターンマッチ	0.824	0.831	0.827
提案手法	0.852	**0.916**	**0.883**

5. おわりに

本論文では，学習者が書いた英文中のローマ字語を自動認識する手法を提案した．実験の結果，提案手法はローマ字語の認識に対して有効であることを確認した．提案手法の利点として，認識対象の文書と英単語リストしか必要としない低コストな手法であることが挙げられる．なお，提案手法に基づいたローマ字語認識ツールを http://www.ai.info.mie-u.ac.jp/~nagata/tools/にて公開中である．

参考文献

[1] J. Aarts and S. Granger, Tag sequences in learner corpora: a key to interlanguage grammar and discourse, Longman Pub Group, 1998.

[2] E. Brill, G. Kacmarcik, and C. Brockett, “Automatically harvesting Katakana-English term pairs from search engine query logs,” Proc. of 6th Natural Language Processing Pacific Rim Symposium, pp.393–399, 2001.

[3] L. Burnard, Users Reference Guide for the British National Corpus. version 1.0, Oxford University Computing Services, Oxford, 1995.

[4] M. Chodorow and C. Leacock, “An unsupervised method for detecting grammatical errors,” Proc. of 1st Meeting of the North America Chapter of the Association for Computational Linguistics, pp.140–147, 2000.

[5] S. Granger, “Prefabricated patterns in advanced EFL writing: collocations and formulae,” in Phraseology: theory, analysis, and application, ed. A.P. Cowie, pp.145–160, Clarendon Press, 1998.

[6] N.R. Han, M. Chodorow, and C. Leacock, “Detecting errors in English article usage by non-native speakers,” Natural Language Engineering, vol.12, no.2, pp.115–129, 2006.

[7] E. Izumi, K. Uchimoto, T. Saiga, T. Supnithi, and H. Isahara, “Automatic error detection in the Japanese learners’ English spoken data,” Proc. of 41st Annual Meeting of the Association for Computational Linguistics, pp.145–148, 2003.

[8] K.S. Jeong, S.H. Myaeng, J.S. Lee, and K.S. Choi, “Automatic identification and back-transliteration of foreign words for information retrieval,” Information Processing and Management, vol.35, pp.523–540, 1999.

[9] B.O. Khaltar, A. Fujii, and T. Ishikawa, “Extracting loanwords from Mongolian corpora and producing a Japanese-Mongolian bilingual dictionary,” Proc. of 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, pp.657–664, 2006.

[10] K. Knight and J. Graehl, “Machine transliteration,” Computational Linguistics, vol.24, no.4, pp.599–612, 1998.

[11] G. Leech, P. Rayson, and A. Wilson, Word Frequencies in Written and Spoken English: based on the British National Corpus, Longman, 2001.

[12] R. Nagata, A. Kawai, K. Morihiro, and N. Isu, “A feedback-augmented method for detecting errors in the writing of learners of English,” Proc. of 44th Annual Meeting of the Association for Computational Linguistics, pp.241–248, 2006.

[13] A.F. Nwesri, S.M. Tahaghoghi, and F. Scholer, “Capturing out-of-vocabulary words in Arabic text,” Proc. of 2006 Conference on Empirical Methods in Natural Language Processing, pp.258–266, 2006.

[14] Y. Tono, “A corpus-based analysis of interlanguage development: analysing POS tag sequences of EFL learner corpora,” Practical Applications in Language Corpora, pp.123–132, 2000.

付　　録

正規化のための置換ルール:

連続した同種の二つの子音 → 一つの子音 （例：$tt \rightarrow t$），

([bdfghjklmnstprz])y([auo]) → \$1\$2 （例：$bya \rightarrow ba$），

([sc])h([aiueo]) → \$1\$2 （例：$sha \rightarrow sa$）

tsu → tu